This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the ICC 2007 proceedings.

A New Relaxation Labeling Architecture for Secure Localization in Sensor Networks

Chih-Chieh Geoff Chang
North Carolina State University
geoff_chang@ncsu.edu

Wesley E. Snyder
North Carolina State University
wes@ncsu.edu

Cliff Wang
U.S. Army Research Office
cliff.wang@us.army.mil

***Abstract*— In this paper, a new strategy is proposed to defend against colluding malicious nodes in a sensor network. The new strategy is based on a new relaxation labeling algorithm to classify nodes into benign or malicious ones. Only reports from benign nodes can then be used to perform localization and obtain accurate results. Experimental results based on simulations and field experiments illustrate the performance of the algorithm.**

I. Introduction

As the applications of sensor networks continue to be developed, the capability to precisely calculate the locations of the objects of interest becomes increasingly important [1], [2]. Most of the localization algorithms can be categorized as either range-based or range-free. Range-based methods collect range reports from sensor nodes and use them to calculate the exact position of the target; while range-free methods instead use hop counts.

The secure localization issue has been investigated only recently. Sensor networks are often deployed in hostile environment, hence vulnerable to a variety of attacks [3], [4] which will lower the accuracy of localization. To defeat such attacks, one strategy is to focus on the nodes before doing the localization task. One may concentrate on verifying a node's claim that it is located inside a physical region [5], or authenticating the nodes and their messages [6]–[8]. Another strategy is to try to localize the events as accurately as possible even under attacks. These methods are based on the concept of majority and let the majority of the nodes to decide. For example, [9] uses medium-square error and [10] uses voting.

In this paper, we create a new strategy against colluding attacks. Section II will define the problem and explain what our contributions are, and Section III will explain our proposed algorithm. Section IV demonstrates experimental result, and conclusion follows afterwards.

II. Goals and Assumptions

A. Problem Statement

Consider the case when we have deployed a sensor network, and all of the node positions have been known and calibrated. All of the nodes are active and listen to the channel for possible event occurrences, and those who have detected the event will report it. An unknown number of nodes are replaced with malicious nodes by the adversary, or some unknown number of additional malicious nodes are introduced into the network. The problem of *secure localization* is to correctly identify the location of the event under the influence of malicious nodes.

B. Localization Model

We assume that sensor nodes only have range capabilities. Due to sensing range, only n of the nodes have detected the event. Each sensor node which has detected the event reports a range δ_i, $i = 1, ..., n$ from its location to the unknown event location. Denoting the location of each sensor node as (x_i, y_i), we obtain

$$(x_i - x_0)^2 + (y_i - y_0)^2 = \delta_i^2, \qquad i = 1, 2, ..., n \quad (1)$$

where (x_0, y_0) is the unknown location of the event. To localize the event in the presence of zero-mean additive Gaussian noise is to minimize the following objective function

$$\underset{(x_0,y_0)}{argmin} \sum_{i=1}^{n} \left[(x_i - x_0)^2 + (y_i - y_0)^2 - \delta_i^2\right]^2 \quad (2)$$

In (1), if n sensor nodes reported the event, we will have n equations. If we subtract the $i = 1$ equation from

1-4244-0353-7/07/$25.00 ©2007 IEEE

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2007		**00-00-2007 to 00-00-2007**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
A New Relaxation Labeling Architecture for Secure Localization in Sensor Networks	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **North Carolina State University,Raleigh,NC,27695**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
See also ADM002055. Proceedings of the 2007 IEEE International Conference of Communications (ICC 2007) Held in Glasgow, Scotland on June 24-28, 2007. U.S. Government or Federal Rights License

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**6**	

the other equations, we obtain $(n-1)$ equations.

$$2(x_i - x_1)x_0 + 2(y_i - y_1)y_0 = (x_i^2 - x_1^2) + (y_i^2 - y_1^2) + \delta_1^2 - \delta_i^2 \quad (3)$$

where $i = 2, ..., n$. Denoting the unknown location of the event as $\mathbf{x} = \begin{bmatrix} x_0 & y_0 \end{bmatrix}^t$, and

$$\mathbf{A} = \begin{bmatrix} 2(x_2 - x_1) & 2(y_2 - y_1) \\ \vdots & \vdots \\ 2(x_n - x_1) & 2(y_n - y_1) \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} (x_2^2 - x_1^2) + (y_2^2 - y_1^2) + \delta_1^2 - \delta_2^2 \\ \vdots \\ (x_n^2 - x_1^2) + (y_n^2 - y_1^2) + \delta_1^2 - \delta_n^2 \end{bmatrix}$$

We get the following linear system of equations

$$\mathbf{A}\mathbf{x} = \mathbf{B}. \quad (4)$$

The location of the event can be found by $\mathbf{x} = \mathbf{A}^{-1}\mathbf{B}$ [1]. If $n \geq 3$, we can use the minimum-squared-error solution $\mathbf{x} = \left[\mathbf{A}^t\mathbf{A}\right]^{-1}\mathbf{A}^t\mathbf{B}$ to estimate the event location [1]. Such localization method is often referred to as *triangulation*, and it is also known in GPS localization [11] and cellular device localization [12]. Note that in triangulation, we need at least 3 nodes which have reported a range to calculate the event location.

C. Problem Definition

For each of the n nodes that have detected the event, the location, (x_i, y_i), and range report δ_i are known. We denote a set of 3 nodes as a *triple*. Since $n > 3$, we have $\binom{n}{3} = \frac{n!}{(n-3)!3!} = \frac{4 \cdot 5 \ldots n}{1 \cdot 2 \cdot 3 \ldots (n-3)}$ triples. For each triple, we can readily obtain an estimate of the event location, $(\tilde{x_0}, \tilde{y_0})$, using triangulation.

There is an unknown number of malicious nodes inside the network. Furthermore, we assume that the malicious nodes are colluding - i.e. their range reports point to a fictitious event location. Within each triple, we can calculate the error terms of the 3 nodes by

$$\epsilon_i = |\delta_i - \sqrt{(\tilde{x_0} - x_i)^2 + (\tilde{y_0} - y_i)^2}|, i = 1, 2, 3 \quad (5)$$

The problem is then to detect the malicious nodes in the network using all the available information.

D. Assumptions

1) The event is static: We assume that the event detected by the sensor network is not moving.

2) Redundancy of the sensor measurements: We assume that when an event occurs, there are more than enough ($n > 3$) nodes which have detected the event.

3) Malicious nodes: We assume that malicious nodes can successfully authenticate with the network, and they also have obtained the encryption key used by existing nodes. Hence malicious nodes can successfully communicate within the network, and any effort to use encryption or authentication to detect them will be futile.

4) Fictitious event: The fictitious event location has to fall within the sensing range of all the active nodes. Otherwise, the benign nodes can simply tell which node is lying.

5) Centralized Model: Since we are using a triangulation scheme, we assume a centralized system in which a central processor will do the triangulation and also process security measures. Secure localization methods for ad hoc systems [13] are not covered in this paper.

E. Threat Analysis

Current literature proposes methods that use a majority concept to deal with malicious nodes [9], [10]. When the attackers are not colluding, these techniques can detect those attackers who are outliers to the community. However, when the attackers are colluding, we argue that they become a consistent group which are harder to detect. It gets even worse when the colluding malicious nodes becomes the majority. Hence our work focuses on colluding malicious nodes.

F. Contributions

Our contribution is two-fold. First, we extend the relaxation labeling algorithm to use higher-order compatibility functions [14] since localization requires triples. Second, we solve the problem of colluding attacks in secure localization in sensor networks. Our strategy is to examine the different behaviors of each triple, and exploit their inconsistency using relaxation labeling. Once the malicious nodes are detected, we can use only reports from benign nodes to perform localization.

III. A New Relaxation Labeling Architecture

Our motivation is that if there is a malicious node in a triple, the report from the malicious one will be different from the other two. On the other hand, if all three nodes in a triple are benign (or malicious), the reports will be consistent. An ideal algorithm to exploit such consistency concept is relaxation labeling [14]–[16], which was proposed first in *image processing*. Suppose that we are analyzing a picture, in which we have detected several objects. In order to label each object unambiguously, we can use the relationships that exist between these objects. However, in classic relaxation

labeling, the compatibility function is often defined for two objects. In the context of sensor networks, we need to design higher-order compatibility functions [14] since we are dealing with triples.

We use a *label* λ for each sensor node as assigning it to be malicious or benign. For example, if a node has label λ_0, it is considered to be malicious, while having label λ_1 denotes benign. For each sensor node, we define a *confidence* $P(\lambda)$. The confidence of node i having label λ_j is denoted as $P_i(\lambda_j)$. The confidence $P_i(\lambda_j)$ has probability-like properties:

$$0 \leq P_i(\lambda_0), P_i(\lambda_1) \leq 1 \qquad P_i(\lambda_0) + P_i(\lambda_1) = 1. \quad (6)$$

Following [15], we will iteratively update the confidence of node i having label j as

$$P_i^{t+1}(\lambda_j) = \frac{P_i^t(\lambda_j)\left[1 + q_i^t(\lambda_j)\right]}{D_i^t}, \quad j = 0, 1 \quad (7)$$

where $D_i^t = \sum_j P_i^t(\lambda_j)\left[1 + q_i^t(\lambda_j)\right]$ is a normalization required to ensure that $P_i(\lambda_j)$ sums to 1, and t stands for iteration.

Next we define $q_i^t(\lambda_j)$ to be a measure of how consistent the labeling of node i as having label λ is with the labeling of the other two nodes:

$$q_i^t(\lambda) = \frac{1}{N} \sum_j \sum_k \sum_{\lambda'} P_j(\lambda') \sum_{\lambda''} P_k(\lambda'') r(\cdot), \quad (8)$$

where $N = (n-1)(n-2)$, n is the number of active nodes in the network, $j = 1, ..., n$, $k = 1, ..., n$, $j \neq i$, $k \neq i$, $j \neq k$, and $r(\cdot)$ is a compatibility function to be defined and explained later. The power of $q_i^t(\lambda)$ is that it considers the consistency of node i having label λ with all other nodes j, k. If node i having λ is not consistent with other nodes, $q_i^k(\lambda)$ will be negative, hence driving $P_i(\lambda)$ down.

Therefore, we need to design a compatibility function $r(i, j, k, \lambda, \lambda', \lambda'')$ which will be higher when nodes i, j, k having label $\lambda, \lambda', \lambda''$, respectively, is consistent, and vice versa. Specifically, $-1 \leq r(\cdot) \leq 1$. For example, if we have 10 nodes, and we are examining nodes 3, 5 and 7. If node 3 is malicious, and nodes 5 and 7 are benign, then $r(3, 5, 7, \lambda_0, \lambda_1, \lambda_1)$ should be close to 1.

For each triple having nodes (i, j, k), we may calculate the respective error ϵ_i by equation (5). Hence we have ϵ_i, ϵ_j, ϵ_k and $\epsilon = \epsilon_i + \epsilon_j + \epsilon_k$. Each node of the triple could be either benign or malicious, so we ought to have $2^3 = 8$ different compatibility functions for a triple. However, we assume that the malicious nodes are colluding, i.e. all the malicious nodes will report that an artificial event occurred at a particular location (which is, of course, different from the true event location). In this regard, the case where all three nodes are benign is as compatible as the case where all three nodes malicious. Let us consider the 3-node benign case. Since all of the 3 nodes are benign, the total error should be small. Hence for the 3-node benign case, we can define

$$r(i, j, k, \lambda_1, \lambda_1, \lambda_1) = 1 - \frac{2}{1 + e^{-\alpha_1(\epsilon - T_1)}} \quad (9)$$

where T_1 is a threshold on the error term ϵ. In equation (9), ϵ is the total error of the triple. The larger ϵ is, the less likely that this triple has all benign nodes, because a large ϵ indicates that the nodes in the triple do not agree with each other. Note that in equation (9), the function of the form $1/(1 + e^{-\alpha(x-\beta)})$ is generally referred to as a *sigmoid function* [17]. The advantage of using a sigmoid function is that the compatibility function will be a smooth curve, and the output is always constrained between -1 and 1, which is what we want for a compatibility function. We show some examples of equation (9) in Figure 1.

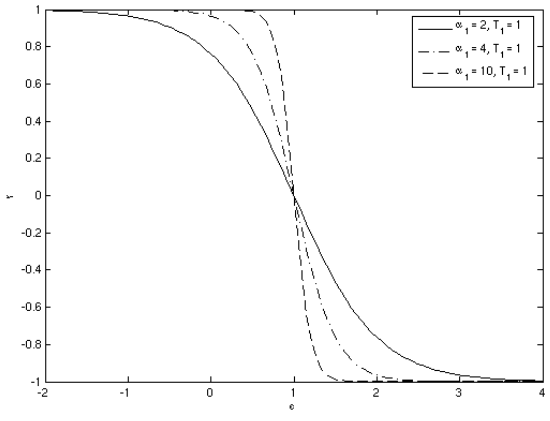


Fig. 1. Some example parameter settings of equation (9)

The compatibility function for the 3-node malicious case is identical to (9) since the malicious nodes are assumed to be colluding.

Next, if there is only one malicious node in the triple, the other two nodes must be benign. This one-node malicious case is exactly the same as the one-node benign case, because in the one-node benign case, the other two malicious nodes are colluding.

Let us look at the one-node malicious case first. For one-node malicious case, the malicious node will tend to have a higher error. If node i is malicious, then we define the compatibility function for one-node malicious (one-node benign) case as

$$r(i,j,k,\lambda_0,\lambda_1,\lambda_1) = \begin{cases} -0.5, & \epsilon > T_1 \\ \frac{1}{1+e^{-\alpha_2(x-T_2)}} - 0.5, & \epsilon \leq T_1 \end{cases} \quad (10)$$

where $x = \epsilon_i/\epsilon$, the ratio of the error by the malicious node i divided by the total error, ϵ. The x in (10) measures the contribution of the malicious node to the total error, ϵ. In equation (10), the larger x is, the higher r is. That is to say, if we find that for a particular triple (i,j,k), the error comes from node i has a significantly large contribution to the total error ϵ, then we have a high confidence to claim that node i is malicious and nodes j and k are benign.

The other 1-node benign/malicious compatibility functions are defined in the same manner as equation (10). Moreover, we lower the return values in (10) to be between -0.5 and 0.5, since empirically we find it to be more stable. We summarize all 8 compatibility functions in Table I. Note that in Table I, $x_\zeta = \epsilon_\zeta/\epsilon$, where $\zeta = i,j,k$. The algorithm of relaxation labeling is summarized in Table II.

$r(i,j,k,\lambda_0,\lambda_0,\lambda_0)$	$1-2/(1+e^{-\alpha_1(\epsilon-T_1)})$	
$r(i,j,k,\lambda_0,\lambda_0,\lambda_1)$	-0.5	if $\epsilon > T_1$
	$1/(1+e^{-\alpha_2(x_k-T_2)})-0.5$	if $\epsilon \leq T_1$
$r(i,j,k,\lambda_0,\lambda_1,\lambda_0)$	-0.5	if $\epsilon > T_1$
	$1/(1+e^{-\alpha_2(x_j-T_2)})-0.5$	if $\epsilon \leq T_1$
$r(i,j,k,\lambda_0,\lambda_1,\lambda_1)$	-0.5	if $\epsilon > T_1$
	$1/(1+e^{-\alpha_2(x_i-T_2)})-0.5$	if $\epsilon \leq T_1$
$r(i,j,k,\lambda_1,\lambda_0,\lambda_0)$	-0.5	if $\epsilon > T_1$
	$1/(1+e^{-\alpha_2(x_i-T_2)})-0.5$	if $\epsilon \leq T_1$
$r(i,j,k,\lambda_1,\lambda_0,\lambda_1)$	-0.5	if $\epsilon > T_1$
	$1/(1+e^{-\alpha_2(x_j-T_2)})-0.5$	if $\epsilon \leq T_1$
$r(i,j,k,\lambda_1,\lambda_1,\lambda_0)$	-0.5	if $\epsilon > T_1$
	$1/(1+e^{-\alpha_2(x_k-T_2)})-0.5$	if $\epsilon \leq T_1$
$r(i,j,k,\lambda_1,\lambda_1,\lambda_1)$	$1-2/(1+e^{-\alpha_1(\epsilon-T_1)})$	

TABLE I
COMPATIBILITY FUNCTIONS

Use range estimates δ_i to calculate ϵ_i, ϵ_j, ϵ_k for all triples;
Initialize $P_i(\lambda_0) \simeq P_i(\lambda_1) \simeq 0.5$;
Set κ to be a small positive number;
while (for all nodes ($\kappa < P_i(\lambda_0) < 1-\kappa$) {
for each node {
  Calculate $q_i(\lambda_0)$ and $q_i(\lambda_1)$ using (8);
  Calculate $D_i(\lambda_0) = P_i(\lambda_0)(1+q_i(\lambda_0)) + P_i(\lambda_1)(1+q_i(\lambda_1))$;
  Calculate $P_i(\lambda_0) = P_i(\lambda_0) \times (1+q_i(\lambda_0))/D_i$;
  Calcualte $P_i(\lambda_1) = 1 - P_i(\lambda_0)$;
} }

TABLE II
ALGORITHM FOR RELAXATION LABELING

IV. EXPERIMENTAL RESULTS

We randomly generate the positions of n nodes in a test field of $[0,10] \times [0,10]$. Among the n nodes, n_1 are malicious. We simulate the sensor measurements z_i using the sensor model [1]

$$z_i = \frac{a}{d_i^2} + w_i, \quad (11)$$

where a is the amplitude of the event, d_i is the distance from sensor i to the event, and w is additive Gaussian noise. We set $a = 1.0$, and equation (12) becomes

$$z_i = \frac{1}{d_i^2} + w_i \quad (12)$$

For benign nodes, d_i is the distance from node i to (x_b, y_b), the location of the true event. Similarly, for malicious nodes, d_i is the distance from node i to (x_m, y_m), the artificial event location that the malicious nodes report. In our simulations, we remove any randomly-placed nodes that happen to be located equidistantly to (x_b, y_b) and (x_m, y_m), because the range reported by such nodes is ambiguous and independent of whether they are assigned to malicious or benign. Note that in (12), d_i is in the order of 0 to 10, so $1/d_i^2$ is in the order of 0 to 0.01. Hence in our simulations, noise variance $\sigma^2 = 0.01$ is considered to be rather large. To obtain range estimates from each sensor node, we calculate

$$\delta_i = \sqrt{\frac{1}{z_i}} = \sqrt{\frac{1}{\frac{1}{d_i^2} + w}} = \sqrt{\frac{d_i^2}{1 + d_i^2 w}} \quad (13)$$

In our first experiment, we set $(x_b, y_b) = (3.0, 3.0)$ and $(x_m, y_m) = (7.0, 7.0)$. All of the probabilities are initialized at $P_i(\lambda_j) \simeq 0.5$. The number of nodes, n, is 7, and the first two nodes are always chosen to be malicious, i.e. $n_1 = 2$. The parameters are determined experimentally as $\alpha_1 = 4.0$, $T_1 = 0.1$, $\alpha_2 = 2.0$, $T_2 = 1/3$. Note that α_1 and α_2 merely control the

slope of the sigmoid function in Equation (9) and (10), and small changes of α_1 and α_2 generally do not have any influence on the simulation outcome. T_2 is also almost always set to $1/3$, hence the two parameters that impact the performance are T_1 and the noise variance σ^2.

To verify the correctness of our relaxation labeling algorithm, we set $\sigma^2 = 0.000001$, essentially noise free. The relaxation labeling process took 22 iterations to converge, and the time it took was less than 1 second on a 1.8GHz PowerMac G5. We plot the (random) locations of the 7 nodes in Figure 2. Also, based on δ reported from each sensor, we plot a circle for each node in Figure 2. In Figure 3, we show the convergence of $P_i(\lambda_1)$, $i = 1, ..., 7$. Since $P_0(\lambda_1)$ and $P_1(\lambda_1)$ converged to 0, node 0 and node 1 are found to be malicious.

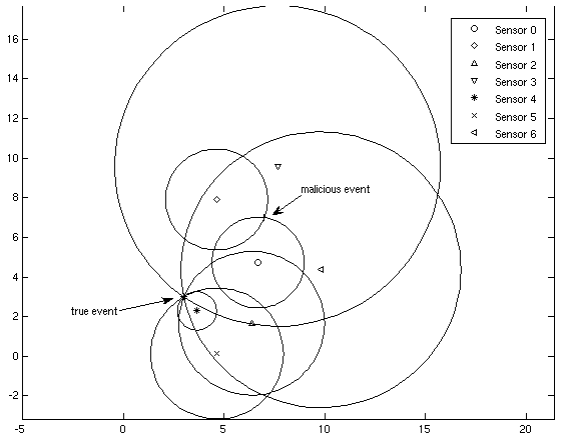


Fig. 2. Simulation Example of 7 nodes. Two of the nodes are malicious.

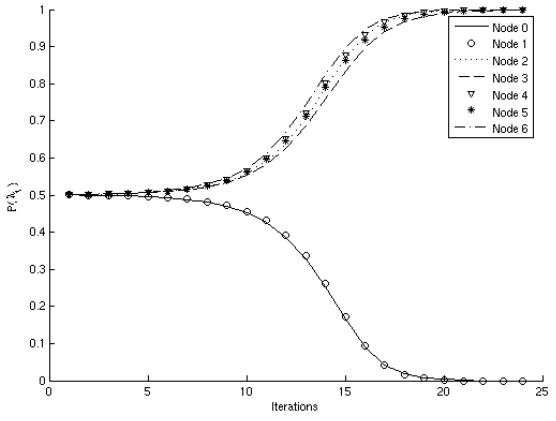


Fig. 3. Convergence of $P(\lambda_1)$

Next, to examine if the relaxation labeling process is correct, we repeat the same experiment using $\sigma^2 = 0.000001$ and $T_1 = 0.01$ for 10,000 times, each with a different set of (random) node locations. The results of detect malicious nodes are all correct.

Next, we look at the impact of parameters on the system performance. Again we choose 2 nodes out of 7 nodes to be malicious, and we repeat the experiment for 100 times. Each time the random locations of the 7 nodes are different, but the parameters, T_1 and σ^2, are the same. For any experiment out of 100 times, we will consider an experiment a failure if ANY one of the 7 nodes is misclassified. Note that this is a very strict failure definition. We then perform another 100-times experiment at different T_1 and noise variance σ^2 values. The failure rate (out of 100 experiment) is shown in Figure 4 as a 3D graph. Note that in Figure 4, the y-axis is log-scale, hence noise variance σ^2 ranges from 1.0 to 0.000001. We can see that at $0.00001 \leq \sigma^2 \leq 0.000001$ and $0.1 \leq T_1 \leq 0.75$, the error rates are almost 0. The error percentage goes up as T_1 goes up. The optimal choice of T_1 is within the range of $[0.1, 0.75]$. As σ^2 goes up, the failure rate goes up considerably. The number of iterations required to reach convergence corresponding to the T_1 and σ^2 values in Figure 4 are illustrated in Figure 5. In Figure 4, the average time to perform any 100-times experiment is 3.489 seconds.

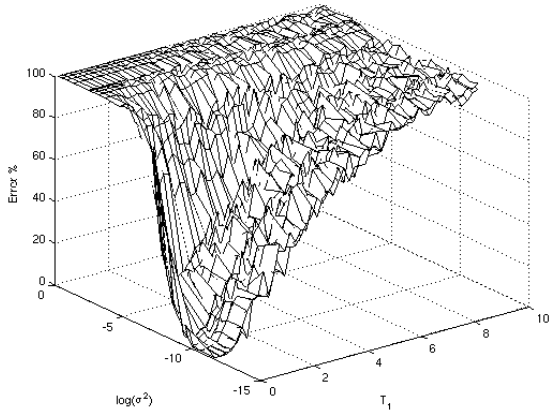


Fig. 4. The effect of T_1 and σ^2 on the system performance

Finally, we fix the number of malicious nodes at 2, and increase the number of total nodes in the network. We also fix $T_1 = 0.5$ and repeat the experiments for 100 times, each time with different random node locations. The failure rate (out of 100 times) is shown in Figure 6(a) at three different noise levels. In Figure 6(a), we can see that the failure rate goes down as the number of nodes gradually increases. We also examine the effect of

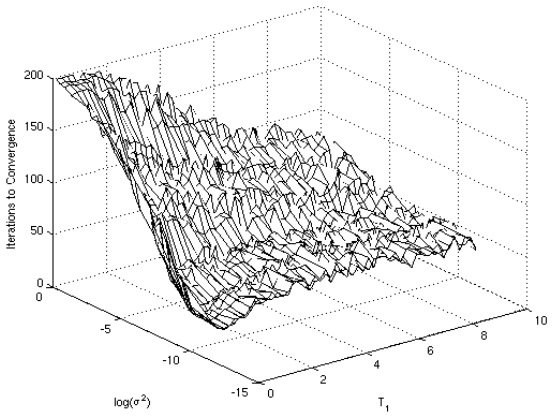
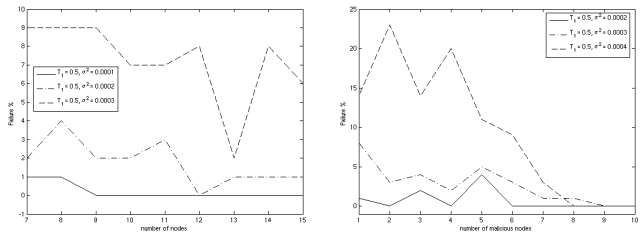


Fig. 5. The number of iterations required to reach convergence corresponding to the T_1 and σ^2 values in Figure 4

the number of malicious nodes on a network in Figure 6(b). In Figure 6(b), we have 20 nodes in the network, and the number of malicious nodes goes from 1 to 19, while $T_1 = 0.5$. Again, we repeat the experiments at three different noise levels. We can see from Figure 6(b) that at noise variance $\sigma^2 = 0.0002$, the system can successfully detect malicious nodes, even with a high number of malicious nodes in the network. Of course, as noise variance σ^2 increases, the system performance goes down. Interestingly, our algorithm seems to work the best when there are about equal numbers of benign and malicious nodes.



(a) (b)

Fig. 6. (a) Failure rate for a network of various number of nodes. At each network size, only 2 nodes are malicious. (b) Failure rate for various number of malicious nodes in a network of 20 nodes.

V. Conclusions

We propose a new relaxation labeling architecture which uses higher-order compatibility functions. Our relaxation labeling algorithm can detect colluding malicious nodes in the sensor network. There are two parameters affecting the performance of our algorithm: the error threshold T_1 and noise variances. With suitable choice of parameters, our algorithm can detect a high number of malicious nodes in the network.

Acknowledgment

This project is supported by United States Army Research Office grant W911NF-04-D-003.

References

[1] F. Zhao and L. J. Guibas, *Wireless sensor networks: an information processing approach*. Morgan Kaufmann Publishers, 2004.

[2] M. Ilyas and I. Mahgoub, Eds., *Handbook of sensor networks: compact wireless and wired sensing systems*. CRC Press LLC, 2005.

[3] A. D. Wood and J. A. Stankovic, "Denial of service in sensor networks," *Computer*, vol. 35, no. 10, pp. 54 – 62, October 2002.

[4] S. Slijepcevic, J. L. Wong, and M. Potkonjak, "Security and privacy protection in wireless sensor networks," in *Handbook of sensor networks: compact wireless and wired sensing systems*, M. Ilyas and I. Mahgoub, Eds. CRC Press LLC, 2005, ch. 31, pp. 31.1 – 31.18.

[5] N. Sastry, U. Shankar, and D. Wagner, "Secure verification of location claims," in *Proceedings of the 2003 ACM workshop on Wireless security*. ACM Press, September 2003, pp. 1 – 10.

[6] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "Spins: security protocols for sensor networks," *Wireless Networks*, vol. 8, no. 5, pp. 521 – 534, September 2002.

[7] T. Park and K. G. Shin, "Lisp: A lightweight security protocol for wireless sensor networks," *ACM Transactions on Embedded Computing Systems*, vol. 3, no. 3, pp. 634 – 660, August 2004.

[8] D. Liu, P. Ning, and R. Li, "Establishing pairwise keys in distributed sensor networks," *ACM Transactions on Information and System Security*, vol. 8, no. 1, pp. 41 – 77, February 2005.

[9] Z. Li, W. Trappe, Y. Zhang, and B. Nath, "Robust statistical methods for securing wireless localization in sensor networks," in *Fourth International Symposium on Information Processing in Sensor Networks*, April 15 2005, pp. 91 – 98.

[10] D. Liu, P. Ning, and W. K. Du, "Attack-resistant location estimation in sensor networks," in *Information Processing in Sensor Networks*, 2005.

[11] Y. Zhao, *Vehicle location and navigation systems*. Artech House, 1997.

[12] A. H. Sayed, A. Tarighat, and N. Khajehnouri, "Network-based wireless location: challenges faced in developing techniques for accurate wireless location information," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 24 – 40, July 2005.

[13] L. Lazos and R. Poovendran, "Serloc: Robust localization for wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 1, no. 1, pp. 73 – 100, August 2005.

[14] R. A. Hummel and S. W. Zucker, "On the foundations of relaxation labeling processes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, no. 3, pp. 267 – 287, May 1983.

[15] A. Rosenfeld, R. A. Hummel, and S. W. Zucker, "Scene labeling by relaxation operations," *IEEE Transactions on Systems, Man and Cybernetics*, vol. SMC-6, no. 6, pp. 420 – 433, June 1976.

[16] J. Kittler and J. Illingworth, "Relaxation labeling algorithms - a review," *Image and Vision Computing*, vol. 3, no. 4, pp. 206 – 216, 1985.

[17] D. H. von Seggern, *CRC Standard Curves and Surfaces*. CRC Press LLC, 1992.